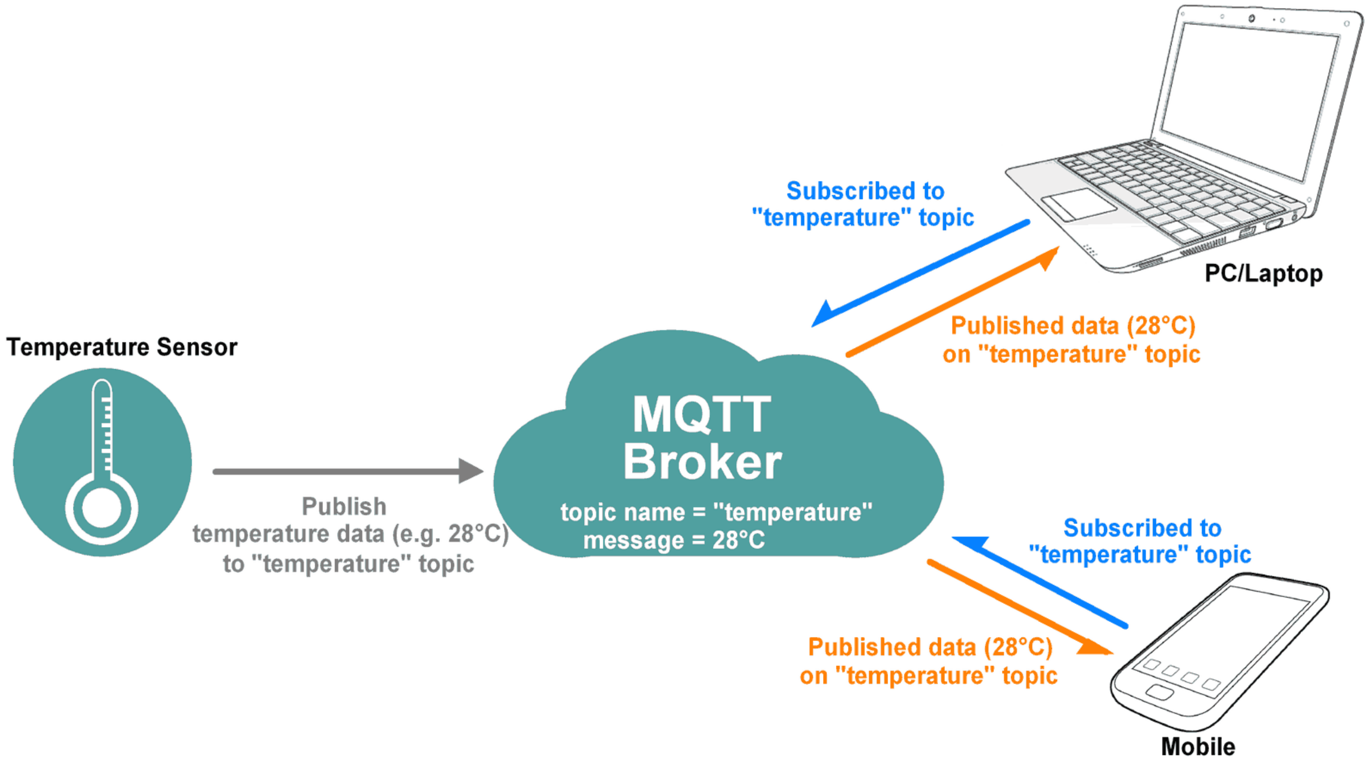
NodeMCU MQTT Client with Arduino IDE

Introduction

MQTT is lightweight publish-subscribe based messaging protocol.

- It is quicker (faster) than other request-response based APIs like HTTP.
- It is developed on the base of TCP/IP protocol.
- It allows remote location devices to connect, subscribe, publish etc. to a specific topic on the server with the help of message broker.
- MQTT Broker/Message broker is a module in between the sender and the receiver. It is an element for message validation, transformation and routing.
- The broker is responsible for distributing messages to the interested clients (subscribed clients) of their interested topic.



For example, if temperature sensor publishes the temperature data (message) on the topic “temperature” then interested clients who have subscribed to “temperature” topic get that published temperature data as shown in above figure.

MQTT is widely used in IoT (Internet of Things) embedded applications, where every sensor is connected to a server and we have access to control them over the internet.

NodeMCU is an open source IoT platform. It is a firmware which runs on ESP8266 Wi-Fi SoC from Espressif Systems. It has on board wi-fi available through which IoT applications becomes easy to build.

The MQTT Client module of NodeMCU is as according to version 3.1.1 of the MQTT protocol. Make sure that your broker supports and is correctly configured for version 3.1.1. let's see the functions used for MQTT on NodeMCU.

MQTT Packet Formation

MQTT uses many packet formats that used to connect to the server and subscribe or publish to the topic on the server.

Refer below link for MQTT OASIS standard. It will help to understand MQTT packet formations.

http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html#_Toc398718027 (http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html#_Toc398718027)

Example

Let's write Arduino program to configure NodeMCU as MQTT Client to

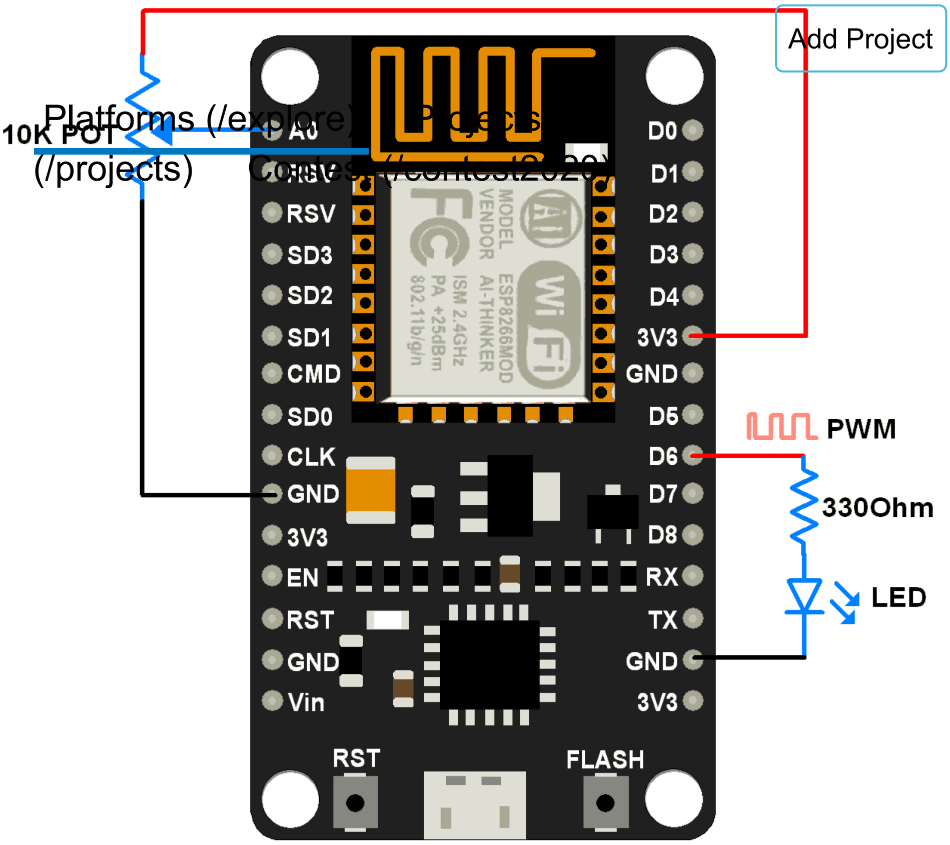
1. Control LED brightness from remote location and
2. Sending voltage across POT(Potentiometer) in digital form to remote location from Adafruit dashboard.

Here we are using Adafruit server for MQTT Client demo purpose.

In IOT platform, Adafruit IO Dashboard allow us to visualize and provides control over connected device to internet. Anyone can visualize and analyse live data from their sensor devices. To learn more and start with Adafruit IO Dashboard refer link https://learn.adafruit.com/adafruit-io-basics-dashboards/creating-a-dashboard (https://learn.adafruit.com/adafruit-io-basics-dashboards/creating-a-dashboard)

Just sign up and create dashboard. After successful creating of dashboard we will get AIO key which is later used to access feed data.

Once we created dashboard on Adafruit we can add various blocks that can be used to control devices as well as monitor status of devices. To see more about blocks, refer link https://learn.adafruit.com/adafruit-io-basics-dashboards/adding-blocks (https://learn.adafruit.com/adafruit-io-basics-dashboards/adding-blocks)



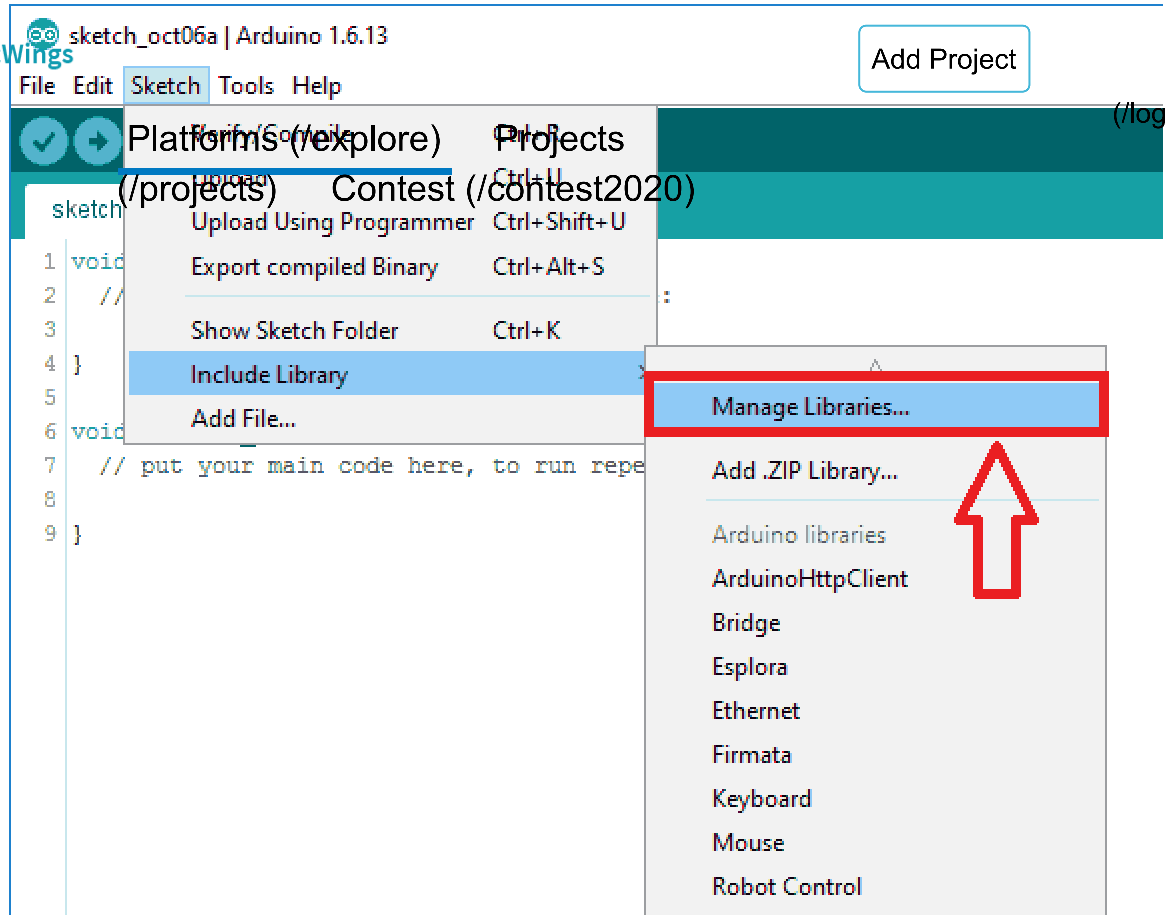
POT and LED connection to NodeMCU

Install required libraries

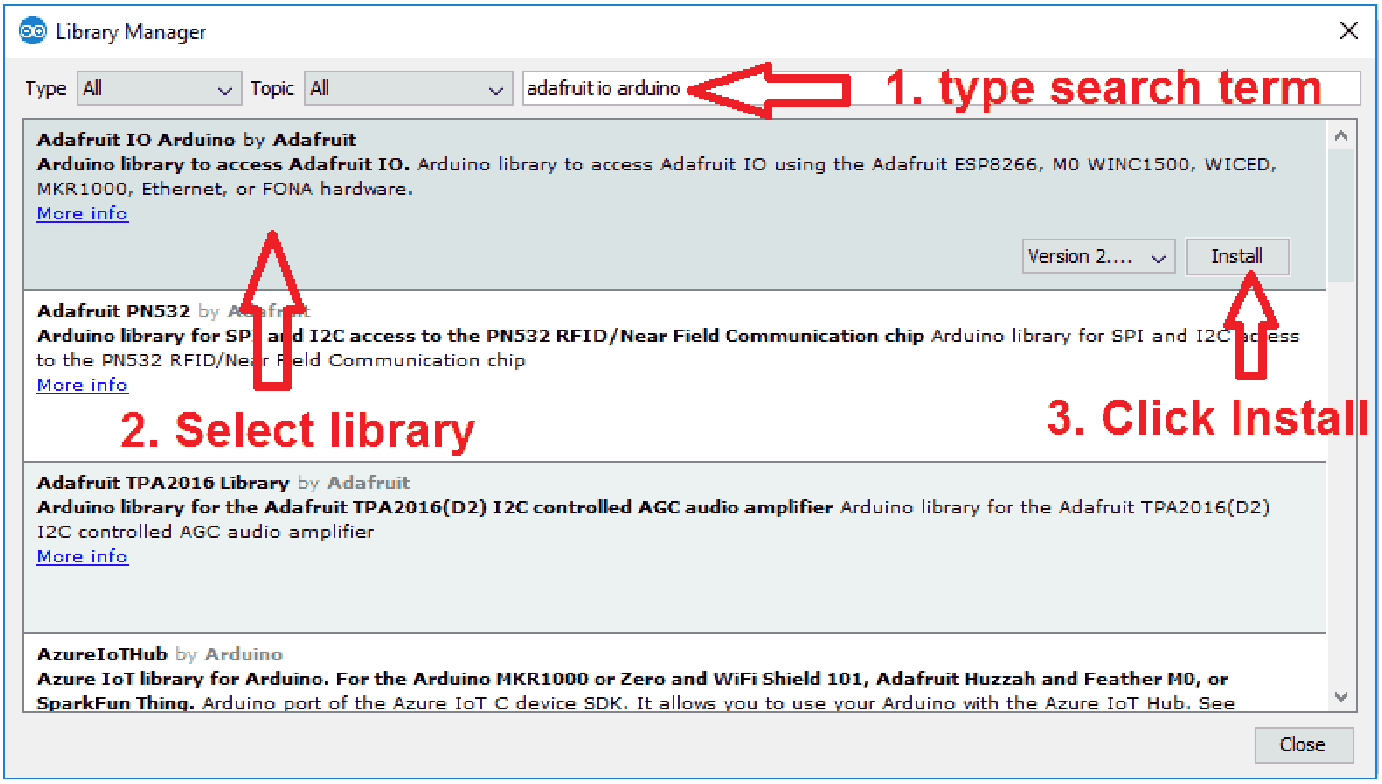
First refer Getting Started with NodeMCU using Arduino IDE (http://www.electronicwings.com/nodemcu/getting-started-with-nodemcu-using-arduino-ide) if you are not installed NodeMCU board packages in Arduino IDE.

Here we are using Adafruit libraries for above example. We will need to install the **Adafruit IO**, **Adafruit MQTT**, and **ArduinoHttpClient** libraries using the Arduino Library Manager.

Open the Arduino IDE and navigate to **Sketch -> Include Library -> Manage Libraries…**



Library Manager window will pop up. Now enter **Adafruit IO Arduino** into the search box, and click Install on the **Adafruit IO Arduino library** option to install version 2.6.0 or higher.



1. type search term

Adafruit IO Arduino by **Adafruit**
Arduino library to access Adafruit IO. Arduino library to access Adafruit IO using the Adafruit ESP8266, M0 WINC1500, WICED, MKR1000, Ethernet, or FONA hardware.
More info

Adafruit PN532 by Adafruit
Arduino library for SPI and I2C access to the PN532 RFID/Near Field Communication chip Arduino library for SPI and I2C access to the PN532 RFID/Near Field Communication chip
More info

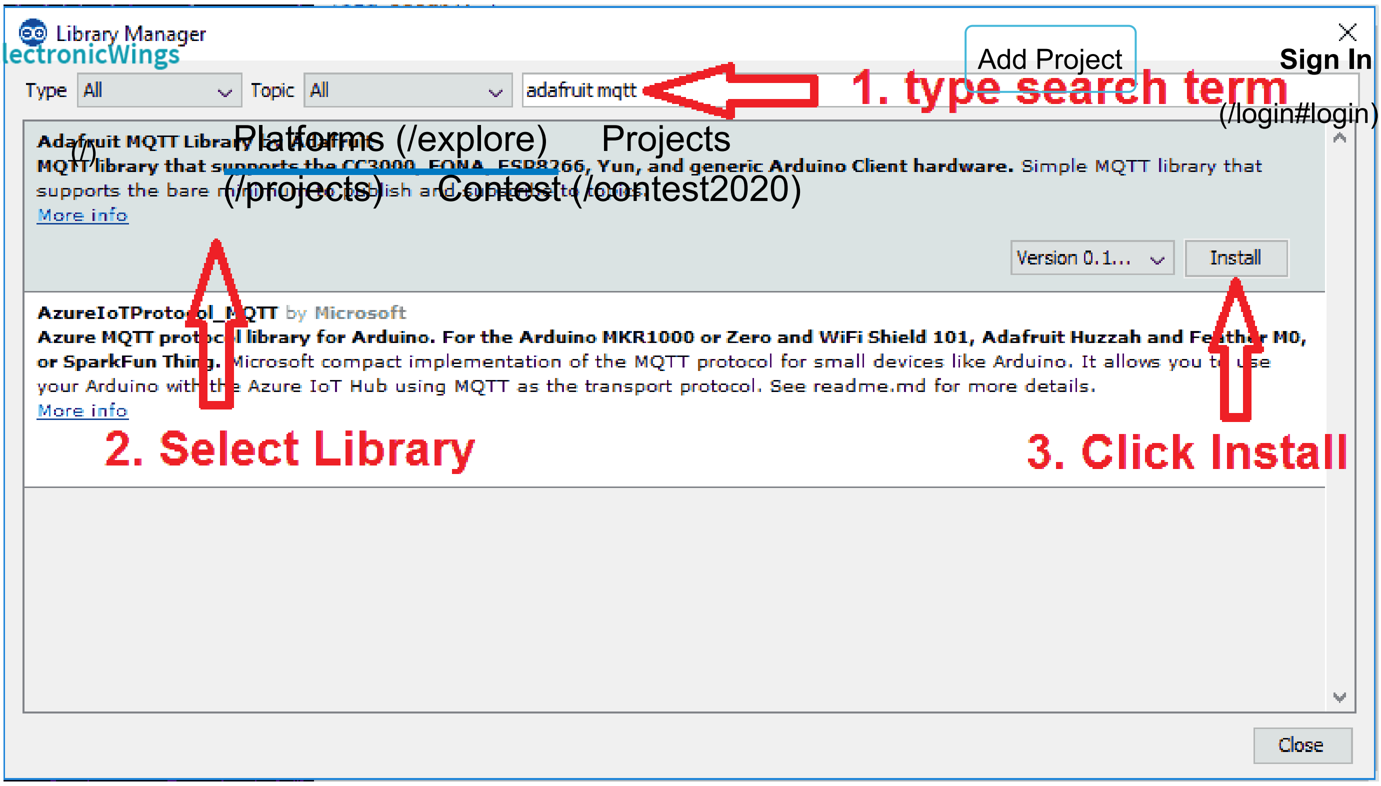
2. Select library

3. Click Install

Adafruit TPA2016 Library by Adafruit
Arduino library for the Adafruit TPA2016(D2) I2C controlled AGC audio amplifier Arduino library for the Adafruit TPA2016(D2) I2C controlled AGC audio amplifier
More info

AzureIoTHub by Arduino
Azure IoT library for Arduino. For the Arduino MKR1000 or Zero and WiFi Shield 101, Adafruit Huzzah and Feather M0, or SparkFun Thing. Arduino port of the Azure IoT C device SDK. It allows you to use your Arduino with the Azure IoT Hub. See

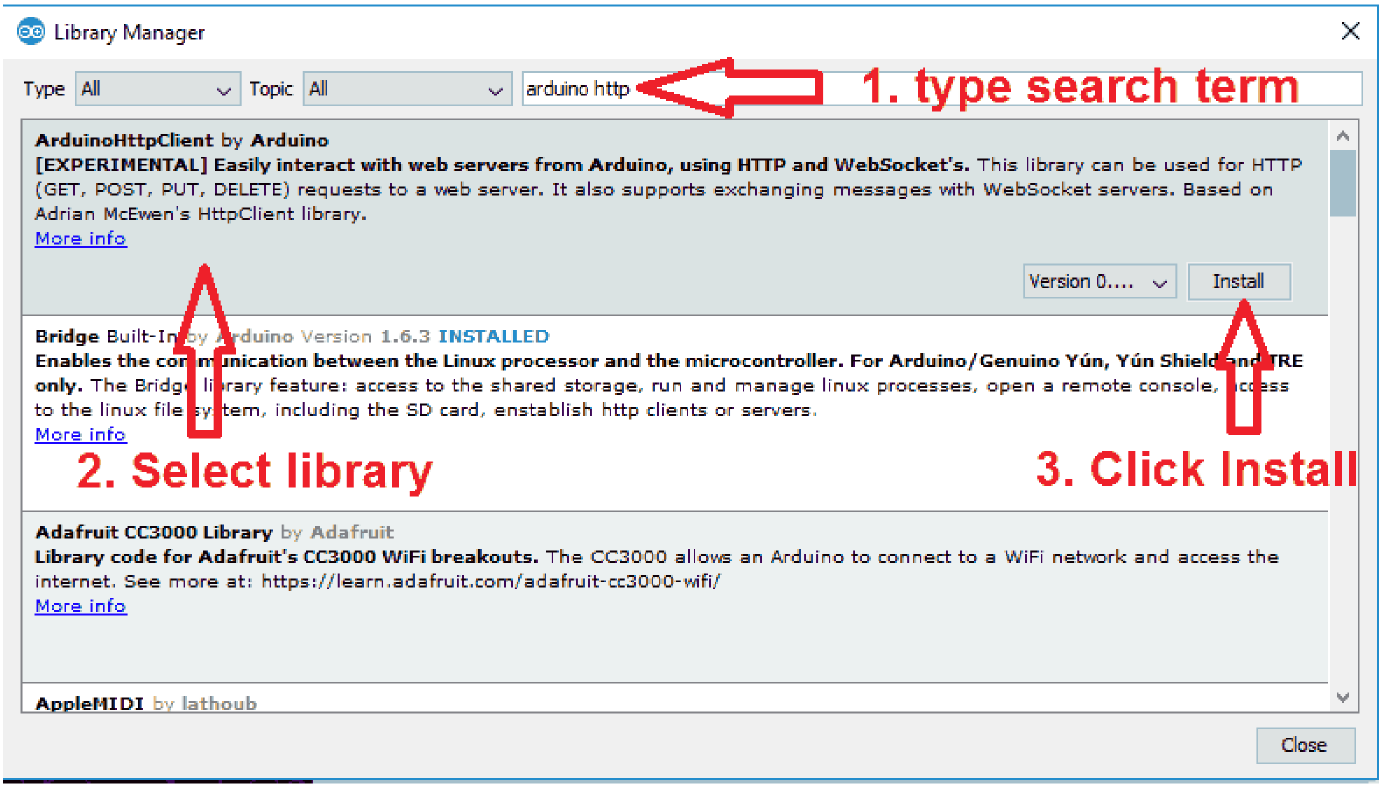
Now enter **Adafruit MQTT** into the search box, and click Install on the **Adafruit MQTT library** option to install version 0.17.0 or higher.



Adafruit MQTT Library by Adafruit
MQTT library that supports the CC3000, FONA, ESP8266, Yun, and generic Arduino Client hardware. Simple MQTT library that supports the bare minimum to publish and subscribe to topics.
More info

AzureIoTProtocol_MQTT by Microsoft
Azure MQTT protocol library for Arduino. For the Arduino MKR1000 or Zero and WiFi Shield 101, Adafruit Huzzah and Feather M0, or SparkFun Thing. Microsoft compact implementation of the MQTT protocol for small devices like Arduino. It allows you to use your Arduino with the Azure IoT Hub using MQTT as the transport protocol. See readme.md for more details.
More info

Now enter **Arduino Http Client** into the search box, and click Install on the **ArduinoHttpClient** library option to install version 0.3.0 or higher.



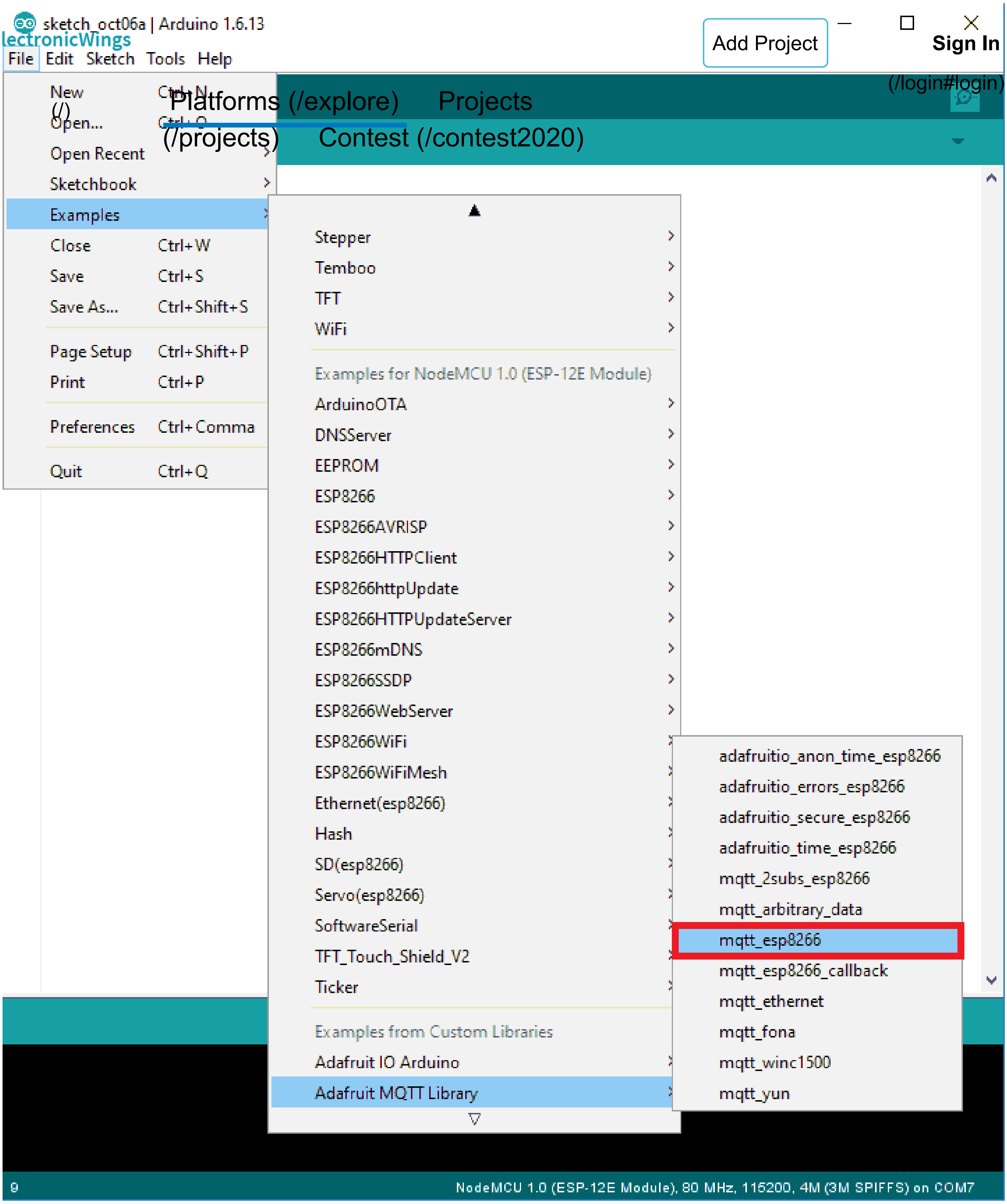
ArduinoHttpClient by **Arduino**
[EXPERIMENTAL] Easily interact with web servers from Arduino, using HTTP and WebSocket's. This library can be used for HTTP (GET, POST, PUT, DELETE) requests to a web server. It also supports exchanging messages with WebSocket servers. Based on Adrian McEwen's HttpClient library.
More info

Bridge Built-In by Arduino Version 1.6.3 **INSTALLED**
Enables the communication between the Linux processor and the microcontroller. For Arduino/Genuino Yún, Yún Shield and TRE only. The Bridge library feature: access to the shared storage, run and manage linux processes, open a remote console, access to the linux file system, including the SD card, enstablish http clients or servers.
More info

Adafruit CC3000 Library by Adafruit
Library code for Adafruit's CC3000 WiFi breakouts. The CC3000 allows an Arduino to connect to a WiFi network and access the internet. See more at: https://learn.adafruit.com/adafruit-cc3000-wifi/
More info

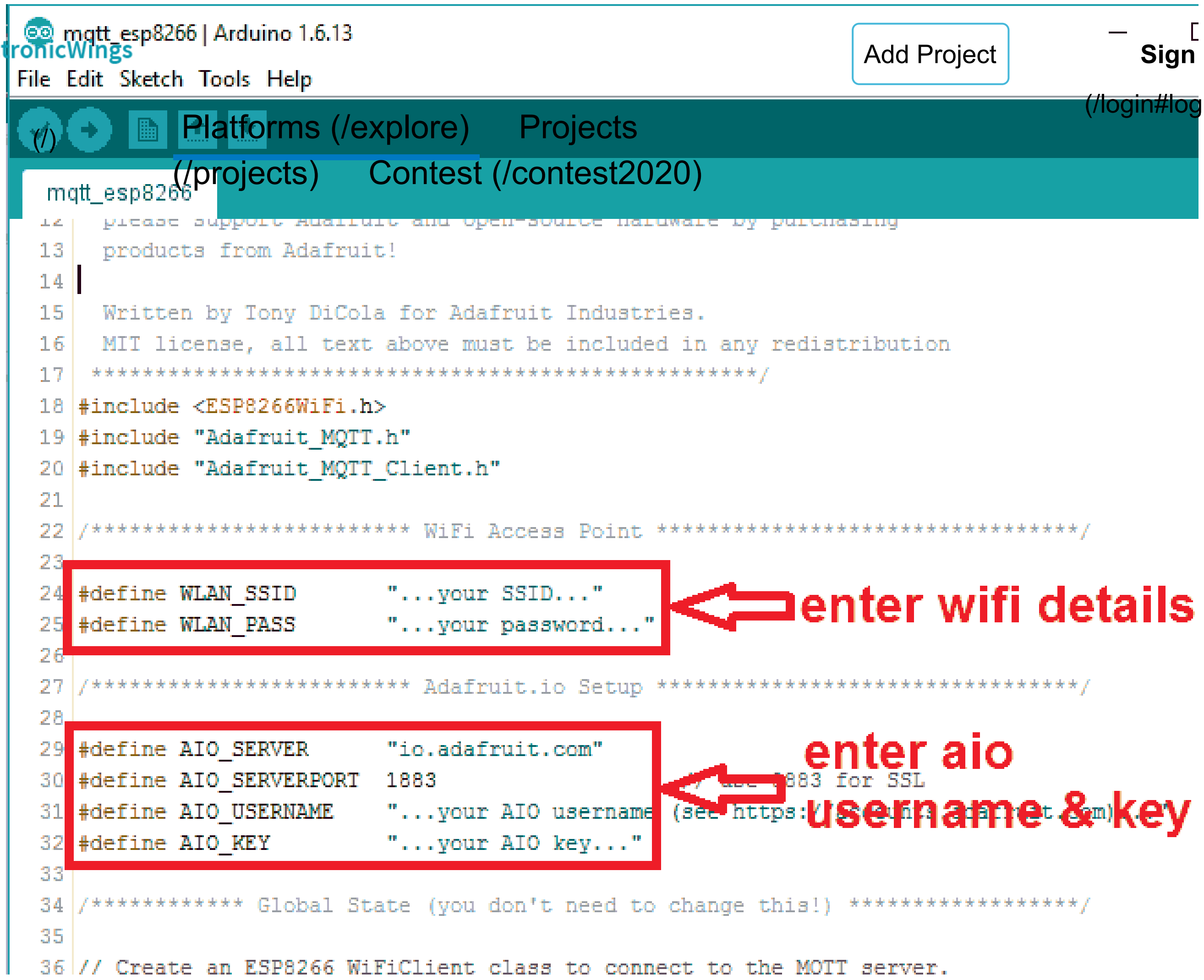
AppleMIDI by lathoub

Now open example of Adafruit mqtt io dashboard. To open it navigate to **File -> Examples -> Adafruit MQTT Library -> mqtt_esp8266**



NodeMCU 1.0 (ESP-12E Module), 80 MHz, 115200, 4M (3M SPIFFS) on COM7

Now edit the wifi and Adafruit io credentials with correct information of example as shown in below image.



```
  please support Adafruit and open-source hardware by purchasing
  products from Adafruit!

  Written by Tony DiCola for Adafruit Industries.
  MIT license, all text above must be included in any redistribution
 ****************************************************/
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

/************************* WiFi Access Point *********************************/

#define WLAN_SSID       "...your SSID..."
#define WLAN_PASS       "...your password..."

/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "...your AIO username (see https://accounts.adafruit.com)..."
#define AIO_KEY         "...your AIO key..."

/************ Global State (you don't need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
```

enter wifi details

enter aio username & key

We have modified mqtt_esp8266 example as per our above example as below

Arduino Sketch for MQTT Client

```
/***************************************************
  Adafruit MQTT Library ESP8266 Example

  Must use ESP8266 Arduino from:
    https://github.com/esp8266/Arduino

  Works great with Adafruit's Huzzah ESP board & Feather
  ----> https://www.adafruit.com/product/2471
  ----> https://www.adafruit.com/products/2821

  Adafruit invests time and resources providing this open source code,
  please support Adafruit and open-source hardware by purchasing
  products from Adafruit!

  Written by Tony DiCola for Adafruit Industries.
  MIT license, all text above must be included in any redistribution
 ****************************************************/
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

/************************* WiFi Access Point *********************************/

#define WLAN_SSID       "...your SSID..."
#define WLAN_PASS       "...your password..."

/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "...your AIO username (see https://accounts.adafruit.com)..."
#define AIO_KEY         "...your AIO key..."

/************ Global State (you don't need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KE

/****************************** Feeds ***************************************/
```

```
// Setup a feed called 'potValue' for publishing.
// Notice MQTT paths for AIO follow the form: <username>/feeds/<feedname>
Adafruit_MQTT_Publish potValue = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feed

// Setup a feed called 'ledBrightness' for subscribing to changes.
Adafruit_MQTT_Subscribe ledBrightness = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERN

/*************************** Sketch Code ************************************/

// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
void MQTT_connect();

uint8_t ledPin = D6;
uint16_t potAdcValue = 0;
uint16_t ledBrightValue = 0;

void setup() {
  Serial.begin(9600);
  delay(10);

  Serial.println(F("Adafruit MQTT demo"));

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());

  // Setup MQTT subscription for ledBrightness feed.
  mqtt.subscribe(&ledBrightness);
}

void loop() {
  // Ensure the connection to the MQTT server is alive (this will make the first
  // connection and automatically reconnect when disconnected). See the MQTT_connect
  // function definition further below.
  MQTT_connect();

  // this is our 'wait for incoming subscription packets' busy subloop
  // try to spend your time here

  Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(200))) {
    if (subscription == &ledBrightness) {
      Serial.print(F("Got LED Brightness : "));
      ledBrightValue = atoi((char *)ledBrightness.lastread);
      Serial.println(ledBrightValue);
      analogWrite(ledPin, ledBrightValue);
    }
  }

  // Now we can publish stuff!
  uint16_t AdcValue = analogRead(A0);
  if((AdcValue > (potAdcValue + 7)) || (AdcValue < (potAdcValue - 7))){
    potAdcValue = AdcValue;
    Serial.print(F("Sending pot val "));
    Serial.print(potAdcValue);
    Serial.print("...");
    if (! potValue.publish(potAdcValue)) {
      Serial.println(F("Failed"));
    } else {
      Serial.println(F("OK!"));
    }
  }
  // ping the server to keep the mqtt connection alive
  // NOT required if you are publishing once every KEEPALIVE seconds
  /*
 if(! mqtt.ping()) {
  mqtt.disconnect();
 }
 */
}

// Function to connect and reconnect as necessary to the MQTT server.
// Should be called in the loop function and it will take care if connecting.
void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
```

Source Code

- NodeMCU MQTT Client

  Download 1487 (/download/attachments=Sat-10-17-19-20-08.NodeMCU_Adafruit_MQTT_IO.zip)

Comments

Please login/signup to comment and reply.

embedprochd (/users/embedprochd/profile)
2019-11-03 19:59:12

my led is not dimming but the pot value is shown on cell phone and comport. But by sliding bar on phone led does not glow .

Reply Like

muhammedimdaad16 (/users/muhammedimdaad16/profile)
2020-07-08 21:11:11

You have to make a feed for subscription

Reply Like

embedprochd (/users/embedprochd/profile)
2019-11-03 19:59:34

please reply soon

Reply Like

Connect On: (https://www.facebook.com/electronicwings) (https://www.youtube.com/channel/UCNdqkukBtk4WhaBqcTsnn3w) (https://www.instagram.com/electronicwingscommunity/?hl=en)

About Us (/about) | Terms of Service (/terms-of-service) | Contact (/contactus)

```
  if (mqtt.connected()) {
    return;
  }

  Serial.print("Connecting to MQTT... ");

  uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
       Serial.println(mqtt.connectErrorString(ret));
       Serial.println("Retrying MQTT connection in 5 seconds...");
       mqtt.disconnect();
       delay(5000);  // wait 5 seconds
       retries--;
       if (retries == 0) {
         // basically die and wait for WDT to reset me
         while (1);
       }
  }
  Serial.println("MQTT Connected!");
}
```

Video

Supporting Files